Pages 1 - 70

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

BEFORE THE HONORABLE WILLIAM H. ALSUP

ANDREA BARTZ, et al, )
)
)
Plaintiffs, )
)
vs. ) No. C 24-5417 WHA
)
ANTHROPIC PBC, )
) San Francisco, California
Defendant. ) Thursday
) January 30, 2025
) 10:00 a.m.

**TRANSCRIPT OF PROCEEDINGS**

**APPEARANCES**:

**For Plaintiffs:**
SUSMAN GODFREY, LLP
1000 Louisiana Street
Suite 5100
Houston, Texas 77002
**BY: JUSTIN A. NELSON, ESQ.**
**COLLIN FREDERICKS, ESQ.**
**ALEJANDRA C. SALINAS, ESQ.**

SUSMAN GODFREY, LLP
One Manhattan West
50th Floor
New York, New York 10001
**BY: JAMES CRAIG SMYSER, ESQ.**

**(APPEARANCES CONTINUED ON FOLLOWING PAGE)**

***Reported By: Debra L. Pas, CSR 11916, CRR, RMR, RPR***
*Official Reporter - US District Court*
*Computerized Transcription By Eclipse*

**APPEARANCES: (CONTINUED)**

**For Plaintiffs:**

LIEFF CABRASER HEIMANN & BERNSTEIN, LLP
Embarcadero Center West
275 Battery Street
29th Floor
San Francisco, California 94111

**BY: DANIEL M. HUTCHINSON, ESQ.**
**REILLY T. STOLER, ESQ.**

LIEFF CABRASER HEIMANN & BERNSTEIN, LLP
250 Hudson Street
8th Floor
New York, New York 10013

**BY: RACHEL GEMAN, ESQ.**

**For Defendant:**

ARNOLD & PORTER KAYE SCHOLER, LLP
Three Embarcadero Center
10th Floor
San Francisco, California 94111

**BY: DOUGLAS A. WINTHROP, ESQ.**
**JOSEPH R. FARRIS, ESQ.**
**ESTAYVAINE BRAGG, ESQ.**
**JESSICA L. GILLOTTE**

LATHAM AND WATKINS, LLP
505 Montgomery Street
Suite 2000
San Francisco, California 94111

**BY: JOSEPH R. WETZEL, ESQ.**

**Also Present:**

**BEN ZHAO**
**APARNA SRIDHAR**

\- - -

**THURSDAY - JANUARY 30, 2025** **10:00 A.M.**

**P R O C E E D I N G S**

---000---

**THE CLERK:** Calling Civil Action 24-5417, Bartz, et al, versus Anthropic PCB.

Counsel, please approach the podium and state your appearances for the record, beginning with counsel for plaintiff No.

**MR. NELSON:** Good morning, Your Honor, Justin Nelson from Susman Godfrey.

With me from Susman Godfrey I have Craig Smyser, and Collin Fredericks. With me -- and Alejandra Salinas.

With me from Lieff Cabraser, Rachel Geman, Daniel Hutchinson and Reilly Stoler.

Your Honor, also in attendance, not a lawyer, independent expert is Ben Zhao, who is the Newbauer Professor of Computer Science at University of Chicago and one of four academics listed in the *Time* AI Top 100 List. Any mistakes are totally my fault and not his.

Thank you, Your Honor.

**MR. WINTHROP:** Good morning. Doug Winthrop from Arnold and Porter.

And I'm here with my colleagues, Joe Farris, Jessica Gillotte, Estayvaine Bragg, and my co-counsel Joe Wetzel from Latham and Watkins.

**THE COURT:** Okay. Welcome to all of you.

All right. So we're here so you all can teach me something about artificial intelligence, at least as it pertains to our case.

I would like to get your suggestion -- I think we've set aside two hours. What are your suggestions on -- procedurally on the best way to proceed, and then we'll just get started on it.

**MR. NELSON:** Thank you, Your Honor. Justin Nelson.

I would suggest -- I think the parties had conferred and we've exchanged slides. I think we both have slide presentations.

I would suggest, and I think we've reserved about 45 minutes for an opening presentation. So, of course, up to Your Honor's pleasure.

I would suggest that we present for 45 minutes. Of course, if there are questions or confusions, please feel free to ask, but we've structured our presentation to be, without interruption, about 45 minutes.

**THE COURT:** All right. And yours is how long?

**MR. WINTHROP:** Similar. Actually, probably -- ours is probably shorter. Ours is probably closer to a half hour, but yeah.

**THE COURT:** Okay. This is all fine, except let me ask. I usually find it very helpful if there is a way to --

when you finish and before you go to your next major point, that I give the other side a chance to critique that point and then we move to the next point.

I don't know enough to say where the break point is, but is that possible here?

**MR. WINTHROP:** I think we'll do whatever you want. I would say ours, at least, is, you know, relatively -- the parts flow relatively smoothly together. But if -- if you -- if it works out to your favor to --

**THE COURT:** Okay. All right. All right. You don't like the idea. We're not going to do that.

All right. We'll start with plaintiff. You get to go first.

**MR. NELSON:** Thank you, Your Honor.

I will say that just having exchanged slides, I can't say there's going to be 100 percent agreement, but I did notice that there was a substantial overlap in terms of some of what we're talking about.

**THE COURT:** All right. Now, which one of these is yours?

**MR. NELSON:** The one in the binder, Your Honor.

**THE COURT:** Okay. So let's -- so you get to sit down and listen. And then you'll get the floor in due course.

Who is going to do the presentation?

**MR. NELSON:** Thank you, Your Honor. That will be me.

**THE COURT:** I should say I know very little about artificial intelligence, and I'm going to have some questions along the way.

I know a fair amount about computers. I know a fair amount from having all these cases over the years. Code, I know a fair amount.

But the actual idea of artificial intelligence and how it works and what it's capable of doing, I'm -- the average citizen knows more than I do.

So I'm not up to speed, and this will be very helpful to me.

**MR. NELSON:** Well, thank you, Your Honor. I hope between the presentations today that you are ready for your PhD or at least your Master's.

**THE COURT:** Good, good, good. I'll apply to the University of Chicago.

Go ahead.

**MR. NELSON:** Thank you, Your Honor. May it please the Court, Justin Nelson from Susman Godfrey representing plaintiffs.

Over the next 45 minutes or so we will go through how a large language model works, how expressive content is critical to the functionality of those LLMs, and why books are especially important.

On the screen is a book called *The Feather Thief* by Kirk

Wallace Johnson, who is one of the named plaintiffs. We will be using the *Feather Thief* as an exemplary book throughout today's presentation.

The allegations in this case are that Anthropic used not just the *Feather Thief*, but hundreds of thousands of other books without permission, including that Anthropic downloaded books that came from a known pirated database and stole copies of Mr. Johnson's and the class's books.

One of those datasets alone, Books3, contains approximately 196,640 books.

We will show how LLMs are designed to mimic human expression and how encoding huge quantities of real human expression from a diverse set of sources --

**THE COURT:** You said LLM?

**MR. NELSON:** Large language models.

**THE COURT:** Okay, got it. Thank you.

**MR. NELSON:** -- is how you do that.

Throughout this presentation we'll refer to this as AI, artificial intelligence, but AI really is a misnomer. It's HI, human imitation.

Artificial intelligence is a predictive model. These models predict the next word based upon the content that they ingest.

And if you look on the screen, you can see what is basically the formula, which underpins today's large language

models. It's that the quality of the model is based on, number one, the quality of the data; and, number two, the quality of your compute. Better data, better compute, means a better model because data and the quality of data are so important.

The "large" in "large language models" is because large amounts of data, large training sets, are the key to an LLMs' ability to mimic natural speech. In fact, one of the founders of OpenAI, Ilya Sutskever, said [as read]:

"Data is the fossil fuel of AI. It was like created somehow and now we use it."

Of course, we know it wasn't "like created somehow." It was created by humans, including our plaintiffs, and the AI systems are trying to mimic it.

Books, in particular, are an incredibly valuable finite resource for AI companies.

The models we refer to as AI models are mathematical models applying various forms of mathematics, statistics and pattern recognition, algorithms, to a massive corpus of human expression and intelligence.

And as this illustration shows, the reason an LLM, like Claude, can generate expressive content is because its mind and encoded the expression from countless works like the *Feather Thief* into its predictive model of human intelligence.

Here we see the first sentence of the *Feather Thief* [as read]:

"By the time Edwin Rist stepped off the train on to the platform at Tring, it was already quite late."

As we'll discuss, the LLM process breaks up that sentence and turns those words into something called tokens, which are numerical representations of those words. And that happens not just for the *Feather Thief*, but for all books and all data in the training set.

We ask Claude, for example: Write me the first two pages of a mystery novel, set on the John Muir trail in the 1970's. And it produced a book called High Sierra.

The reason Claude can do that is because it's processed and encoded many, many books, its pattern matching and conditional probabilities. And without that high quality human data, the result is a low quality model; garbage in, garbage out.

How does Claude produce something like High Sierra? We'll be talking about how these LLMs work, from its background through the intricacies on how it encodes expression, to the importance of quality expressive content to the model.

First, let's talk about the background of modern LLMs, where they came from and what they are.

A large language model is exactly what it purports to be. It's large. It works with human language. And it's a type of statistical predictive model.

It is, in short, a statistical model of human language

built upon a neural network architecture. Neural networks were first proposed in the 1950's.

These large language models are a type of neural network that predicts what word comes next based on the words that came before it, but they are much more sophisticated models than from prior generations.

LLMs, predicting one word at a time and stringing those words together, are able to draft and Anthropic's telling everything from a text message to a screenplay to a novel.

Recall the High Sierra example above. It will write you the rest of the book if you prompt it to. And if you tell Claude, "Tell me the first two paragraphs of a *Tale of Two Cities*," as the slide shows, it will do it.

Claude has the ability to accept entire books as input and then predict the next word.

Claude has a context window -- that's the box where the user can put in information -- equivalent to about 150,000 words, or approximately a 500-page book.

The classic example of this capability is to imagine a mystery novel where the final sentence is "The murderer is." If a modern LLM were predicting the last word of that novel, it would be able to take into account every word of the mystery novel in predicting who the murder was, even clues from the first pages of the book.

LLMs do this because they have taken in a huge amount of

human expression. The most recent generation of LLMs are estimated to have approximately 9.75 trillion words.

Performance of today's LLMs are governed by so-called scaling laws. In their simplest form these scaling laws, which are really just observed empirical relationships, mean that the resulting size and performance of these models is dependent on proportional and exponential increases in the amount of data and the amount of specialized computers and chips called graphic processing units, or GPUs, used in training the model.

These scaling models can be seen in the following three graphs. With the "X" axis, an exponential scale. The graphs show that model performance scales with increases in compute and data size.

So these models absorb vast quantities of human expression and then use math to generate probabilistic estimates of the next words.

According to the Complaint, and by public reporting, it was OpenAI that fired the starting gun of this generation of LLMs by creating a proprietary dataset of pirated books which they used to train the models which would go on to power what is now OpenAI's ChatGPT.

Seeing OpenAI's success, according to the Complaint, Anthropic decided that it needed to catch up and did the same. So it downloaded something called the "pile," which contains a dataset of pirated books, named Books3.

Here is the tweet announcing the launch of Books3. Suppose you wanted to train a world-class GPT model, just like OpenAI. How? You have no data. Now you do. Now everyone does. Presenting Books3, a/k/a all of Bibliotik, which is a known pirated website. 196,640 books in plain text.

Next we'll talk about how an LLM is trained and how it works. There are essentially three steps in the training process.

Step 1 is acquiring the data. Step two is pre-training, where the data acquired in Step 1 is used to generate the model. At a high level, pre-training involves taking the data acquired in Step 1 and chopping it up into word fragments known as tokens represented by a number.

The model then takes those tokens and matches them to a long series of numbers called vectors. Vectors act like longitude and latitude coordinates to help locate the token relative to other tokens, the vectors and code information about the token and about the surrounding context in which the token appears.

The model then predicts what the next token will be based on that vector information in millions of iterations. If its prediction is correct, then the model increases the strength of the predictive pathways that led to that prediction. If it is incorrect, the model decreases the strength of the pathway that led to the prediction.

At the end of the pre-training, in all of those iterations, you have a pre-trained model which is called a next token predictor. Meaning, it is literally predicting what the next token will be. In other words, what the next word will be.

The final step, fine-tuning, is how the pre-trained model becomes the chatbot that users can interact with. The company achieves this by showing a model example conversations curated for certain characteristics, or it implements so-called guardrails, like a copyright guardrail.

That's a high level of review of what we're going to talk through.

So how does an LLM encode the human expression in books into a mathematical model? A helpful analogy to understand how a model estimates patterns from its training set is a path through the woods.

Imagine a dense forest. People enter on the left and exit on the right. At first there are no paths, but as people travel through the forest, paths start to take shape. The ones that are more commonly traveled get more worn down.

Over time it becomes clear how to navigate the different areas of the forest. You walk where the people before you walked.

That's what an LLM is doing. It ingests its training set and records how to get to the correct result.

The pre-training step is where the model is trained to mimic human expression. This training occurs on an architecture called a neural network, as shown on the slide. A neural network is a computational model consisting of layers of interconnected nodes, also known as neurons.

This image shows the architecture of a neural network. Specifically, something called a fully-connected, feed-forward neural network, which is also known in the AI jargon as a multi-layer perceptron, or an MLP.

The network is divided into distinct layers marked by, on the slide, those vertical dotted lines with connections flowing from left to right.

Each circle represents a neuron or a node, and the lines between them represent the weighted connections.

Starting from the left we have first the input layer marked as "i." This is where raw data enters the network. It has "n" input neurons labeled as Input 1 through Input n. There can be thousands of input neurons, but in LLMs they generally correspond to the number of unique tokens in the token vocabulary, which in modern LLMs are approximately 100,000, give or take.

Next the hidden layers, marked as $h_i$, $h_2$ through $h_n$, are the network's processing centers. This particular network shows multiple hidden layers demonstrating what is called deep learning. Each neuron in these layers connects to every neuron

in the previous and subsequent layers. These hidden layers process the input data through increasingly complex representations. These connections each carry a weight that the network adjusts during training in response to patterns in the data.

Because these models have billions of neurons and even more connections, the models need huge quantities of these graphic processing units, the GPUs, and huge quantities of data.

Those connections between nodes are like the paths through the woods in our analogy. The more people walk down a path, the more wore down it gets. And the more the training data traverses certain sequences of words or certain expressions, the better the model gets at predicting what word should come next.

The model's weights measure the strength of the connections between the different nodes.

The output layer, marked as "o," is where the network produces its final results. It has "n" output neurons labeled as Output 1 through Output n.

LLMs use a special neural network architecture called a transformer, which processes an entire string of words at once instead of one at a time. This allows it to encode a wealth of information about the text it is trained on by recording how words appear next to each other.

Put simply, transformer architecture is really the combination of two relatively straightforward features.

An encoder is a series of algorithms that processes input sequences in parallel to generate numerical representations of context.

A decoder is another series of algorithms which processes these embeddings in parallel to generate output predictions.

And to be clear, Your Honor, this doesn't have anything to do with the legal conception of transformativeness. And, in fact, the authors who coined the term "transformer" said they chose "transformer" because they liked the sound of the word.

The math underlying this architecture is really pretty simple, relatively speaking. Essentially it's just matrix multiplication and repeated application of the chain rule from calculus. Math, which is in the grasp of a bright high schooler.

Although I hope Your Honor will forgive me if I'm unable to perform calculus off the top of my head this morning.

It is the prevalence of matrix multiplication and the training process which is why GPUs are so highly prized. It turns out that video game graphics use a ton of matrix multiplication as well.

At a high level, the pre-training process essentially consists of the model attempting a fill-in-the-blank quiz over and over again and adjusting each time based on whether the

model's prediction was right or wrong using a function called back propagation. That function is really just repeated application of the chain rule.

Here is a simple example of the prediction process. Take, "One small step for man, one giant leap for," blank. In this made-up example the model might predict "penguin," "mankind" or "frogs." We know if it picks "penguins," it's wrong. It checks to see that "mankind" is the right answer, updating it the next time.

Let's talk about the four main steps for how it gets to that point this time using the *Feather Thief* as an example.

First, tokenization. Second, embedding. Third, contextual encoding. And fourth, prediction. Each one of these four steps happens millions of times to train a single model.

A single cycle through the four steps is what is called an iteration. And each time, at the prediction phase, the model is checking its prediction against the underlying text in its training data for accuracy and to improve its predictive capacity. If the model training process is functioning properly, the model should continue to improve each iteration.

So let's start with tokenization. After the training material is acquired, it needs to be tokenized so that the training set can be put in a format that the model can ingest.

As I said earlier, "token" means a word or part of a word

represented by a number. "Tokenization" is almost a translation, but not quite. But because there is a set relationship between a given word and a given token, tokenizing a particular work will always generate the same tokens in the same order.

For example, here is OpenAI's public tokenizer. You can see the sentence, "Many words map to one token, but some don't: Indivisible," in that highlighted first.

Now, that's -- according to the tokenizer, in the green box there are 53 tokens. And that first sentence -- if you go to the bottom part of that slide, that first sentence, "Many words map to one token," shows you where exactly each token is.

Most, you see, Your Honor, consist of the entire word, but the comma, for example, is its own token. The colon is its own token. And ironically "indivisible" is not indivisible. It has two tokens.

The very last on the way down is a string of numbers, and you'll see that 123 is its own token. 456 is a separate own token. And 789 are also its own token.

That's in text form, but we can convert it to tokens. That's the exact same thing, except instead of words it's in tokens. What was previously text has been turned into numerical tokens, but every time the same words are used it will produce the same sequence of tokens.

So here, again, is the first sentence from the *Feather*

*Thief* [as read]:

"By the time Edwin Rist stepped off the train onto the platform Tring, 40 miles north of London, it was already quite late."

You see that up top.

Down below you see in the OpenAI public tokenizer that converts to 29 tokens, and you see where the token breaks occur.

And then, if you change that text to token IDs, it becomes numbers. And every time that sentence appears, it will have that same set of numbers.

Also *Feather Thief* is a 300-page book and every sentence of that book, every part of the training set would be broken down into tokens.

The second step is converting those tokens into vectors, which is something called embedding. The concept of embedding traces back to the 80's and 90's. The idea is a word is known by the company it keeps. So instead of just one number, the vector is a series of numbers that acts like a longitude and latitude coordinate for that token in multi-dimensional space.

To start, the vectors are essentially random, although they improve as a model trains. Here on the right we're looking at five words as they might appear as vectors. But, of course, in the actual model these wouldn't be the words themselves. They would be the tokens that have been converted

from the words.

The segment passes through what's called the embedding layer, which converts each token into a numerical vector. So you see, take the word "by." It goes through the embedding layer. And then the top vector, "by" is converted to, in this example, 23, 52, negative 10, et cetera. And because it's multi-dimensional, that vector will go on for a number of other characters as well.

Now, when the model begins the training process, the vectors are essentially random coordinates. But throughout the training process, running those four steps over and over, the vectors are going to be updated over and over again until they start to encode mathematically information about each word conditional on the surrounding words in context.

These conditional probabilities mean that over time and after running through these four phases over and over again, each word in the vector space will end up being close to similar type words.

For example, here "king" would end up being close to "queen" and near "prince." And the difference in vector space also encodes mathematical representations of the words.

For example, "king" minus "man" might equal "royalty." "King" minus "man" plus "woman" might equal "queen."

And "king" minus "queen" might approximately equal "man" minus "woman."

And the vector difference between "man" and "woman" is similar to the vector difference, in this example, between "king" and "queen."

Returning to our chart of the steps, now we're at Step 3, which is contextual encoding. Step 2 was converting the tokens into vectors. Step 3 is where those vectors gets updated based upon how those words appear next to each other in the training data.

The contextual encoding step is the key difference between modern LLMs and what came before.

At Step 3 the vectors are fed into something called an attention block which adjusts each token's vector based on the surrounding tokens. For each word it looks at all of the surrounding words and identifies which ones are relevant to updating its representation of the word in question. Then it moves around the word's coordinate based on the surrounding context.

Critical to all of this is the context for how the words appear. For example, in the sentence, "By the time Edwin resist stepped off the train onto the platform at Tring, 40 miles north of London, it was already quite late," the attention block would update its vector for platform based on the word "train." We're not talking about a political party's platform. We're talking about a train platform.

Similarly, it would also update its encoding of Tring

based on the phrase that follows, "40 miles north of London." We're not talking about the video game Monster Tring. We're talking about the town 40 miles north of London.

Another example would be the word "plant." In general, you might think of, well, a plant. But in a thriller novel you might think of a spy. The attention block updates based on how words appear next to each other.

And next, each vector passes through a multi-layer perceptron block. This multilayer perceptron block is a version of the neural network we showed earlier, with inputs, hidden layers and outputs.

In this step the model checks each word in isolation for pattern matching. Remember, all of this is in service of an ultimate prediction. The model will be predicting what word comes next.

So, for example, in predicting the next word it is looking like -- words like "time," "already," and "quite." You can think of this modeling, what part of the forest it's in. The system is modeling based on the expression in its training set and how those words were used, which words are likely to impact its prediction more and which are not.

Take our example sentence. The words "time," "already," and "quite" are all important clues for what word will come next. Whereas, words like "the" have less impact on the prediction.

If instead there were surrounding words like "negligence," "breach," "plaintiff," the model would recognize that those words are legal words and, therefore, would be more likely to predict other legal words to come next.

The model then adjusts the vectors again based on those patterns. That process then repeats as the vectors pass through additional attention blocks and multi-layer perceptron blocks adjusting each time.

To be clear, in Step 3, even in one of these millions of iterations, the sentence is passing through lots of attention blocks and multi-layer perceptron blocks with its vectors updating each time.

And now we're at the last step, prediction. At Step 4 the model applies simple calculus to those updated vectors to generate a probability distribution over all tokens in the vocabulary from which a new token can be probabilistically selected.

Here is the example sentence again [as read]:

"By the time Edwin Rist stepped off of the train onto the platform at Tring, 40 miles north of London, it was already quite blank."

In this example that we are making up, the model might predict "dark" as the most likely next word, with words like "late," "chilly," "crowded," et cetera, after that.

This is conditional probability. The model is calculating

the probability of the next word based on all of the words that came before it. Here the model predicted "dark." The model did not predict "dark" in a vacuum. It predicted "dark" because of all the words that came before it.

The probability distribution, the model's prediction, is compared to the right answer, the actual next word in the text in its training data and the accuracy or inaccuracy of the distribution is then calculated.

So here in the *Feather Thief* the actual next word was "late," not "dark." "Late" was the model's second most likely prediction in this hypothetical example.

At every single step the underlying expression, in this case the book, is the critical input and provides the answer key or in AI lingo something called the ground truth.

How --

**THE COURT:** The what?

**MR. NELSON:** The ground truth.

**THE COURT:** G-R-O-U-N-D?

**MR. NELSON:** Correct, sir.

How mathematically good the prediction was is quantified through something called a loss function. If the model returned a high probability for a correct answer, then the guess was pretty good. If the model returned a low probability, then the guess was pretty bad.

The loss function is measuring how well the model fits the

training set, and it's that training set that provides the correct answers.

After quantifying how good or how bad the guess was, the previous steps are run in reverse and that back propagation function, which is the repeated application of the Chain Rule, adjusts the weights at each layer. The weights that pointed in the right direction gets amplified, while the weights that pointed in the wrong direction get downplayed.

Picking back up on our path through the woods analogy earlier, this is like the model retracing its steps over the path that led to the correct destination. This back propagation is how the model updates from the training data.

By repeating this process over and over again in millions of iterations the model becomes better and better at taking text and outputting a probability distribution that predicts the next word in the training set.

At the end of the pre-training phase, the model is a next token predictor calibrated to its training set. Pre-training results in what is called a base model.

Base models then typically go through another round of training called fine-tuning, where human engineers intervene to train the model to provide certain types of responses and not other types of responses.

In fine-tuning some AI companies build in so-called guardrails. For example, if you try to get Claude to spit out

copyright information, you might notice that it resists. That resistance is not anything inherent to LLMs or to Claude. Instead, it's a gloss grafted on by, in this case, Anthropic or other AI companies.

The reason Claude and other AI companies resist is because of constraints that accompany a company like Anthropic has at some point put on its models to attempt to prevent regurgitation of the copyrighted materials in its training set.

And, in general, there are three types of constraints. The first is something called user side blocking, which is before the prompt goes into the model. This guardrail looks at words in the user's prompt and sanitizes the input so that the model doesn't output infringing or other inappropriate materials.

Second is something called alignment training, which is changing the model based on human feedback. The most common form of this is where humans rate responses as good or bad and the model updates based upon that.

Third is something called post-interference filtering, which looks at the model's output as it is being generated and blocks it if it's bad.

Without those constraints regurgitation would happen all the time. We can see that by how easily the model reproduces non-copyrighted works, like the Bible or a *Tale of Two Cities*. The fact that companies need to use these constraints is a

further example that the model does copy its training data.

Now, we've talked about how LLMs are created. Let's talk about how they work in practice.

How does an LLM, like Claude, work when a consumer is using it? This stage is called inference, as opposed to what we have been discussing up until now, which was training.

We saw at the very beginning of this presentation an example of inference. A user says: Write me the opening two pages of a mystery novel set on the John Muir trail in the 1970's, and the model drafts one.

Here is the inference process. And as you see, it looks a lot like the training process.

This explanation of inference will go pretty quick because the inference process, indeed, is almost identical to the training process. That is, when consumers use Claude and the model outputs a response, it's essentially following the same process we just described.

The difference is that the input to the model comes from the user rather than the training set, and the model isn't updated based upon its predictions at the end.

So at Step 1, that prompt to write a mystery will be tokenized.

At Step 2, those tokens will be converted into vectors.

At Step 3, those vectors are updated based on what the model saw in its training set during pre-training and the

context of the user's input, which, as we said, in the context window can take up to about 150,000 words.

The key difference is at Step 4. The model still predicts a new word, but instead of having its prediction checked against the underlying human expression contained in its training set, it simply outputs the word to the users.

Now, the AI company can also adjust how often the model spits out the most likely next word every time or something lower on the probability distribution by adjusting something called the temperature. The lower the temperature, the more likely it is to spit out the most highly predicted next word.

It's important to note that the model is doing pattern recognition of how words appear next to each other. So sometimes it gets things wrong.

You can see this, for example, in what's known as hallucinations. Large language models will often generate content that looks extremely credible and accurate, but in reality it's completely made up. A false matching to patterns.

So if you ask OpenAI's ChatGPT for case law, it may provide very accurate looking case citations which are completely false.

A lawyer in New York, for example, submitted a brief that cited the following cases. None of these cases were real. The citations looked real, but not a single one of those cases actually exists.

This brings us to our last point, which is why training data, in particular in this case books, are important to LLMs.

In real estate it's location, location, location. In AI it's data, data, data. And that's because the model is only as good as its inputs. If you want your LLMs to produce well-structured text of expressive variety, it needs to see well-structured expressly varied text during training, a lot of it.

And if you want your LLM to produce text that is written in modern vernacular and not 19th century English and produce text that incorporates modern developments you'll need in copyright works. The ability of an LLM to convincingly mimic speech is a function of the quality of the data and the training set used to train it.

As one OpenAI researcher put it, the "it" in AI models is the dataset. He says: "Models" -- on the screen [as read]:

> "Models are truly approximating their datasets to an incredible degree. Model behavior is determined by your dataset, nothing else. Everything else is a; means to an end in efficiently delivering compute to approximating that dataset."

Books are an especially valuable source for the training set. Article after article has confirmed this.

So, for example, here is a 2024 paper stating that [as read]:

"Books contribute to the models' training by exposing them to a diverse array of textual genres and subject matter, fostering a more comprehensive understanding of language across various domains."

This 2023 paper from researchers at MIT, Google and OpenAI states [as read]:

"The best performing domains comprise high-quality (Books) and heterogeneous (Web) data."

Similarly [as read]:

"Princeton AI researchers concluded that using 'long books as long-context data' was crucial for long-context performance of the models."

These are just a few of the papers.

And, of course, Books3 was created for a reason. Its creators wrote in the 2020 paper introducing the dataset that the pirated books were included [as read]:

"Because books are invaluable for long-range context modeling research and coherent story telling."

Other sources of LLM training set data, blog posts, Wikipedia articles, websites, don't offer the same cohesion over as long a stretch of text.

And this is --

**THE COURT:** Say that last sentence again? Other what?

**MR. NELSON:** Sources of the training set for large language models, like blog posts, Wikipedia articles. They are

going to be shorter, so they are not going to have the context over tens of thousands of words to see how things fit together in long coherence.

So, for example, when it's running through that multi-layer perceptron block, it's going to have the whole narrative put together.

So, and that's why, for example, in Books3, we're looking at something that gives you the long-range context modeling research and the ability to have coherent storytelling, which is why they put it in the dataset.

So this is -- so, for example, think back to how the attention block works, right, where the model learns from how words appear next to each other in context.

In our example we showed a single sentence being fed into the model, but that, of course, was simplified because modern large language models are able to process not just a single sentence, but entire books all at once, learning from those connections across the entire book.

And we can also ask Claude the question. We did. We asked Claude about the importance of books in training data, and here is part of his response -- its response, I should say [as read]:

"Books are indeed a particularly valuable source of training data for several reasons.

"High signal-to-noise ratio. Books generally

represent carefully crafted, edited, and curated content. Unlike social media posts or informal web content, books typically go through extensive review and editing processes, resulting in higher quality information and expression."

Second [as read]:

"Complex reasoning and extended arguments. Books allow authors to develop sophisticated arguments and ideas over hundreds of pages. This extended format enables deep exploration of topics and complex chains of reasoning that shorter formats can't support."

Third [as read]:

"Rich contextual relationships. Books often contain complex networks of references between concepts, characters, and ideas. This interconnected nature provides rich semantic relationships for learning."

**THE COURT:** You say that Claude --

**MR. NELSON:** Yeah. We --

**THE COURT:** Claude itself gave that answer?

**MR. NELSON:** That was directly from Claude, Your Honor.

**THE COURT:** It's pretty cool.

Okay. I have a question at some point, but are you near the end?

**MR. NELSON:** In fact, my next words are "in conclusion," Your Honor.

**THE COURT:** All right. Go ahead.

**MR. NELSON:** In conclusion, if there's one thing to take away from this presentation is that LLMs wouldn't be able to do what they were to do if not for the high-quality training sets, like books.

The entire reason the model works in these four steps that we just went over is because of the high-quality content used to train them.

At every step in the training process the model relies on copies, and it is the expression in that content that makes the model what it is.

So you might have heard of the Infinite Monkey Theorem, that a monkey randomly pressing keys on a typewriter would eventually type out the complete works of William Shakespeare purely by chance.

Well, a few months ago some mathematicians wrote a paper where they analyzed this problem, and they came to the conclusion that even if you had 200,000 monkeys randomly pressing one key per second until the universe ends, you could not actually reproduce Shakespeare's works. In fact, you couldn't even get to *Curious George*.

If LLMs were just generators of random words strung together, like hypothetical monkeys typing away, they wouldn't

be worth much attention.

But the reason LLMs work and the reason we're here is because it's not a bunch of monkeys in the back room. It's humanity's collective expression. And the model, this so-called AI, is engaging in HI, human imitation.

Thank you, Your Honor.

**THE COURT:** I've got a question for you, and the other side can answer it, too. You repeatedly said that the idea is to predict the next word.

**MR. NELSON:** Yes.

**THE COURT:** So how does it know when the sentence ends, and how does it know when whether it's grammatically correct?

**MR. NELSON:** That is all from the training set. It's literally -- it's -- it's predicting the next word, which is -- it's called a next token prediction; right? So it's actually predicting the next token, which is converted back into the word.

But, remember, the token is like there's a dictionary effectively between converting the token and converting the -- to the actual word that it outputs, but it knows to make it grammatically correct.

It knows if you --

**THE COURT:** How does it know -- does it know that it has to have a verb and a --

**MR. NELSON:** It does that through going through these millions of iterations and going through these multi-layer perceptron blocks so that it knows, for example, that a noun and a verb go together.

If you remember back -- let's see. If we can go to the slide that has it broken down into the tokens, which I think is slide --

**THE COURT:** There was one for the comma, one for the period.

**MR. NELSON:** One for the comma and --

**THE COURT:** I got that, but these are only predictors, and they are going to be -- statistically on a given page there's going to be some sentences that are not -- they are sentence fragments.

How does it -- does it have a -- after it -- does it have a check to see, okay, this one doesn't have a verb. We need to put a verb in there. Or how does it solve that problem?

**MR. NELSON:** Well, because -- I mean, it uses thousands and thousands of these GPUs and goes through these months-long training sessions and millions of iterations to get that; but if it's still, after the base model, having issues, that's when the fine-tuning comes in.

And so when it spits out something that might not look correct --

**THE COURT:** All right.

**MR. NELSON:** -- the researcher will -- the human will go in and say: Hey, this one is missing a verb or something. But it --

**THE COURT:** Wait a minute. But you don't want humans to do it. You want Claude to do it.

Okay. I have a different question. I can see the point about predicting the next word, but how does it predict plot?

You know, a good writer would have tension. You know, Actor A and Actor B, there's a chapter in the book that Claude is writing and a good chapter will have tension.

Do you know what I mean by that? In other words, they are fighting over something or disagreeing over something. Or chapter one and two and three are setting up the plot, but the plot slowly unfolds, and then there's a twist in the plot.

Now, how does it predict that kind of more subtle, more graphic general organizational level?

**MR. NELSON:** Because it is seeing hundreds of thousands and millions of books. And it recognizes that when you're asking it to write a novel, that's what a novel is.

**THE COURT:** But you can't possibly do that if it's only predicting the next word. It has to be able to predict the plot. Somehow it has to do that.

**MR. NELSON:** Correct, Your Honor.

But built into that is the very different weights that come back to it. So it's based upon the context that you're

inputting.

So when you're doing, it knows -- remember the example the part of the forest it's in? It knows that it's in the novel part of the forest. So even though it's predicting the next token -- and it does do that.

For example, let's go to slide eight. Slide eight is our High Sierra mystery novel. Okay? And if you actually -- you can -- Your Honor has it on the slides. You can actually zoom in on it and it actually --

**THE COURT:** It's too small. I can't read that.

**MR. NELSON:** In the back you'll see it. It actually creates something that tries to give tension.

And it says this -- you know, the -- I'll just read it for Your Honor [as read]:

"The first sign something was wrong was the ravens."

Right?

And it goes on [as read]:

"Three of them, wheeling in tight circles above the tree line..."

And then it just -- it talks about that; right?

And so it knows that that's what a mystery novel is supposed to do because it's seen mystery novels; right?

And when we asked it to put in something about -- in the 70's, you'll see how it talks about the -- you know, things

that are related specifically to -- you know, look at the second to the last paragraph [as read]:

"My freehand instinctively went to the holster at my hip. In '76, most rangers still carried revolvers."

So it's setting it in the place and time knowing that's what it did. And it does that because even though it's a next word predictor, next token predictor --

**THE COURT:** No. In the '70s I don't think they carried revolvers. Wilderness rangers, I don't think they did then. Law enforcement rangers, of course, they did.

But it's -- what I can read there is pretty good, but I don't -- I'm not -- you're not convincing me -- not convinced.

I'm not understanding how a -- what is the next word likely to be. Could predict what is the next twist in the plot likely to be.

That is, to me, a higher level of -- maybe they -- maybe they do that. Maybe they have got the universe broken out into, you know, many different plots and they can follow one path through the -- through the woods that way.

Okay. Other side gets their say.

Thank you. Excellent presentation. Thank you.

**MR. NELSON:** Thank you, Your Honor.

**THE COURT:** Who at the table over there helped prepare the slides and all that? Raise your hand if you were involved.

**MR. NELSON:** Mr. Frederiks and Mr. Smyser were key. They were absolutely critical to this.

So thank you, Your Honor.

**THE COURT:** Thank you. Thank you.

**MR. NELSON:** I to say, I was -- my -- I was boning up on my calculus and, you know --

**THE COURT:** I don't think some of those were calculus. I think some of those were some other kind of math, but --

**MR. NELSON:** The Chain Rule -- the Chain Rule was -- I was proud of myself for getting the trivial answers right, Your Honor.

**THE COURT:** You did good.

Okay. Next side.

**MR. WINTHROP:** Thank you, Your Honor. Doug Winthrop from Arnold and Porter for Anthropic. And I neglected to tell you our client is here, so I --

**THE COURT:** Where is your client? Go ahead, introduce your client.

**MR. WINTHROP:** Yes. Aparna Sridhar is here, in the back, in-house counsel at Anthropic.

**THE COURT:** Good morning.

**MR. WINTHROP:** Before I start, I want to just point out one thing that just jumped out at me about the presentation and even that last discussion.

There's no claim that the High Sierra essay that you saw

infringed anybody's copyright, that it was substantially similar to anybody's copyrighted work. And that's going to be a pretty important concept as we go through this.

So you'll see a -- a fair amount of overlap between our presentation and theirs, but there are some differences and some things where I think they have misstated and overstated. So I'll point those out as I go through this.

**THE COURT:** Sure. Go ahead.

**MR. WINTHROP:** So I'm going to cover five major areas. As you now know, large language models, LLMs, how they work.

What happens during pre-training.

The idea of fine-tuning, which counsel covered.

And then what Anthropic calls constitutional AI and guardrails.

And then finally we'll talk as well about inference; this process of how the models run, how they actually work. So there will be similarity in the topics that we cover.

So in this case when we're talking about Anthropic's artificial intelligence tool, we're talking about a large language model. So that's where we need to start.

So what is a large language model? In plain language, a large language model captures patterns and statistical relationship information about training data.

And, when prompted, generates data with known uncertainty.

And so breaking that down, it's a mathematical system, as

you've heard. Training data, vast amount of training data is used. It's fed into the system.

The structure of the system allows it to iteratively extract statistical information about the data, such as how likely one word is to appear next to other words or how likely a words is to appear in the beginning, the middle, the end.

And then that kind of probability information is captured within this mathematical system.

So then you say: Okay. What does the LLM do with the information?

When the LLM is prompted to give a response, it can generate that response in natural language by predicting an appropriate next word in a sequence based on the statistical information it has captured from the training data.

What I said is important in this context, by predicting an appropriate next word. I didn't say predicting a precise or specific or the correct next word. And that's important.

If you look at the slide, this concept of known uncertainty. That means there's a range of possible appropriate answers, and the LLM picks among the most likely appropriate answers based on the statistical information it has captured.

It does not always pick a particular answer. It does not always pick -- there's no, like, one correct answer it's always going to get.

And I'm going to cover that in a second, but the implication of what you just heard is that this was very deterministic; that essentially that the model is looking for what's the right answer.

And as I'm going to cover in a second, that's actually not a good sign, a sign -- not a good aspect of a model. A model that does that actually needs to be adjusted. We'll come to that.

So key points about the overall principles that I want to cover that are set out right now.

So, as we said, an LLM learns patterns about -- and relationships within data rather than storing contents.

And related to that, the responses of an LLM don't come from receiving stored text, but from this predictive and probabilistic process.

Responses are based on the patterns and relationships the LLM learned from the data.

So any notion that an LLM is storing data or storing training data and using training data that's stored, that's just fundamentally not true.

And then after the model is trained -- and this is what I was talking about -- it gives varied responses to similar user prompts based on probability.

And I'm going to show you this. In fact, if you give the exact same prompt to the same LLM twice, you're likely going to

get different responses.

It's different than the computer coding that I grew up with, where you put something in. You put the right -- the code language, and out comes a deterministic response. That is not this world.

So use cases for LLMs. An LLM's ability to craft natural language responses based on statistical information that's extracted from the training data gives LLMs a wide range of functions.

And here are just examples of some. So an LLM can write code. It can be used by writers to come up with ideas for potential advertising slogans or ideas for an essay on national parks.

As I mentioned, it can come up for -- interesting for an idea of a High Sierra thriller type mystery. But, again, no claim that in this case anything like that has infringed anybody's copyright.

Healthcare professionals can use an LLM to analyze medical resources and interpret complex clinical data --

**THE COURT:** Go to the first one, software developers. So you're -- I'll give you -- I'll just make up an example. So could you say to Claude: Draft -- using the language you make up -- let's say Python. Using the language Python, write a program in code that can play checkers. And then it would go and do that and deliver you some code?

**MR. WINTHROP:** What's that?

**THE COURT:** Would it then deliver you executable code?

**MR. WINTHROP:** I don't know the answer to that, Your Honor. What I do know is that it can create code.

So what I don't know is whether that means it can create sub-aspects of code to do certain functions or whether it can put it all together in that --

**THE COURT:** Okay.

**MR. WINTHROP:** Yeah.

**THE COURT:** All right.

**MR. WINTHROP:** And I wanted to -- on this notion, the theme of an essay, right, that we talked about this High Sierra. It's also interesting to me that when you looked at that presentation and the *Feather Thief*, the theme of the *Feather Thief* throughout the presentation, still there was no allegation, no -- no output, nothing that said that our client's LLM or anybody's LLM had produced something that was substantially similar.

**THE COURT:** What if you said to Claude -- what's the name of that book?

**MR. WINTHROP:** *Feather Thief*.

**THE COURT:** *Feather Thief*.

**MR. WINTHROP:** The *Feather Thief*.

**THE COURT:** All right. If you use that title, say: Give me the words for that book. Would it do that, or would it

be --

**MR. WINTHROP:** It would not. And I'm going to show you with an example like that. You would get a prompt from Claude that would say something like -- because I can't tell you the exact words because it's not deterministic, but the model has been trained to recognize it as something and say: I'm sorry, I can't do that. That is copyrighted.

**THE COURT:** Okay, all right.

**MR. WINTHROP:** Okay. So let's go on then to how an LLM is built. Here are the key elements that you -- I'm just going to run through them.

Blank model architecture.

Data collection and assembly.

Tokenization.

Pre-training.

And fine-tuning.

Some of the similar steps that counsel just mentioned.

So blank model architecture. That's the fundamental design and structure of the model. The basic code is written. The objective of the model is set out. But the model can't really do anything useful because it hasn't been trained. It's received no training data from which it can start to learn.

So we go to the next part of the building a model, building an LLM, data collection and assembly. So you collect, assemble. You clean up the data before it's ready to be used.

For example, you want to get rid of duplicates. You want to get rid of noise, like, HTML and XML tags and web pages.

And LLMs are trained on a variety of data, because as the -- what the model is trying to do is extract statistical information about data and to do this well, to do it accurately, it needs to be exposed to a wide variety of data, see the vast expanse of how language is used.

So to name just a few, right, LLMs are often trained on portions of common crawl, which is meaning historical snapshots from the internet. Certain source code repositories. Books. Research papers. Government documents. Even court decisions.

Now, there's some interesting things to note about how training data is used. So training data is typically introduced to a model in a variety of orders. So you might say why? Why would that matter? And that's so that the LLM does not become biased and weigh training data presented first more heavily than training data presented later.

So as an analogy, if you read all of Shakespeare's sonnets before reading any science texts, you might temporarily think that all writing, including scientific literature, should be written in iambic pentameter. So that's why there's different ordering of data.

Also, the number of times a model sees each piece of training data is typically limited, and that's to avoid causing the LLM to pay too much attention to any one piece of data.

Again, the goal is to have a broad diversity of many types of data, so you don't want the model to see the same data over and over again.

Another thing that's just interesting to note, which we thought we should share, is just some of the unique aspects of this. So introducing a diversity of training data is important for building a good model in both intuitive and non-intuitive ways.

So, for example, Anthropic has learned that training on source code also makes the model better at logical reasoning. And training on data in one language also makes the model perform better in other languages.

**THE COURT:** Okay.

**MR. WINTHROP:** All right. So now we get to pre-training. As you heard earlier, we know what pre-training is. It involves processing massive amounts of data to create a foundation for language understanding. And the term "language understanding" means that the model has extracted patterns and statistical relationship information from the training data, which patterns and statistical relationships are reflected in the weights in the model.

So let's go here. And you'll -- there's a similarity in certain aspects of how -- how we present because obviously there's some fundamental features that everyone agrees on with the LLMs.

So tokenization. I said a few times, right, that the LLM extracts statistical information about words -- where they go in the sentence, how they relate to other words in a sentence -- and that's to conceptualize what's going on here.

In fact, the model, as you heard earlier, is not extracting statistical information from words. It's extracting statistical information from tokens; right? So the whole training process turns tokens into numerical representations.

And so tokenization is the process of converting input training data into these smaller units represented by numerical values in order to make the training data, such as natural language, processible by the --

**THE COURT:** Let me ask you a question about that. I understand the word gets turned into a token, to a number, but is the number more or less random or do the numbers correspond to something?

**MR. WINTHROP:** I believe that there is a set dictionary of token identifiers, and I think that -- we agree it's about 100,000.

So the tokens then get -- that is how they get identified; right? So if you -- there is this finite list of tokens.

**THE COURT:** Are all the ones, say, in the 25,000 series, are they -- have some relationship or --

**MR. WINTHROP:** Yeah, I don't know. Within the 100,000, how do they order them? I don't know. And that's

something we could find out.

**THE COURT:** Okay.

**MR. WINTHROP:** Okay. And just like as you saw before, right, there are more tokens than words. A token can be part of a word. It can be, in some cases, punctuation. And as I mentioned, there are about 100,000 in this token -- in the token dictionary.

Okay. So you heard about vectors and vectorization this morning. So you have the training data broken down into tokens. Now the model starts to learn how those tokens relate to other tokens, and those relationships are captured numerically through vectors.

So as training progresses, the model assigns literally thousands of numerical values. We tried to represent by this graphic to show just how many, the quantity of data that is assigned to each token, each representing some aspect of the token's relationship to other tokens.

So one can think of it this way with an array of numbers, or you could look at it, think of it, you know, graphically in this other way we show.

Next slide.

And this is a three-dimensional diagram. Each dimension representing some aspect of meaning. So, for example, what these pairs have in common.

But this is a three-dimensional diagram. In an actual

model, there might be literally 10,000 dimensions represented as -- in these vectors. So representing such concepts as time, period association, subject/object relationship, noun, verb, what other token is normally to -- with to form a word, where it goes in a sentence, does it introduce a question. All this data related to the tokens.

So then you get to the critical role of weights.

**THE COURT:** Of what?

**MR. WINTHROP:** Weights. So if you look at the chart here, what we've tried to show, as a human receives more and varied information, right -- for example, words have different meanings based on context, what's around them. They start to learn patterns and relationships.

So that's the same with pre-training a model. In fact, the term "neural network," which you heard this morning, is often used to describe the architecture of a functioning LLM because an LLM is comprised of a series of algorithms that are modeled, in a sense, on the interconnectedness of the human brain.

So as the model is exposed to more and more training data, the thousands of numbers reflecting a vector for a token are processed by the model. The weights, mathematical parameters within the model, are altered as a result of being exposed to the vectorized training date from all the other tokens.

And the model that has not been pre-trained, the weights

are set at random levels.

But then as the model is exposed to ever more training data and, therefore, ever more relationships between tokens, the weights adjust to actual values that reflect the model's growing appreciation for the ever more precise information about the relationship between tokens and how tokens, or as words or parts of words, are actually used in natural language.

So if you look on the screen, Your Honor, this is obviously a simplification. And we're not seeing in this visualization the matrix multiplication and other processes that are occurring which caused the weights and vector values to adjust.

We think it's a helpful visual for what's happening inside an LLM as it's exposed to tokens that have different meanings or usage based on context and the surrounding tokens.

So it's shown here -- just go back one.

If you show here every time the model sees a slightly different use of "cold," including uses demonstrating sarcasm, metaphor -- let's go slowly -- homonym, right, it makes a subtle adjustment in the vector for that token as well as in the model weights.

And we -- we show on here, just graphically; right? It's kind of analogous to the brain learning about different contexts, different uses. Only in the context of an LLM this is done mathematically, as the weights and vectors are adjusted

based on more and more training data.

So that's -- when you finish that process, you have a pre-trained model, but you're far from having anything particularly useful.

So the next topic is fine-tuning. So if you -- let's go to the next one.

The quality of the generated output can be improved by giving the model rewards to guide its behavior during this fine-tuning process.

So, for example, here, if you have the user prompt: How cold is San Francisco in July? A pre-trained model response might be: San Francisco has a temperature of 66 degree Fahrenheit for July average. You know, it's functional. It kind of gives you what you need to know. It's not written very well.

But once a model is fine-tuned, you can get a more -- train the model to give a more fulsome response, something that we -- there's often referred to as a useful assistant. How would this be useful to someone? So that is what goes on in pre-training.

So there are two common modalities for fine-tuning. What's referred to as supervised learning and reinforcement learning.

So with supervised learning, the model is exposed to preferred responses to sample inputs. So, for example, instead

of the nine word answer to the prompt: How cold is San Francisco in July? The model is shown a fuller, more discursive response, and that one would -- again, you'd to get from a helpful assistant. The model is trained on input and response pairs like this.

It's like studying by looking at practice tests and ideal responses prepared by the professor.

Another modality is called reinforcement learning, and that's somewhat different. There the model's responses to certain inputs are judged and the model is trained so that it sees its -- if its response was good or bad. And the analogy would be it's like taking practice questions and then being told immediately whether your answer is good or not that good.

And so through that process you get a more sophisticated model that can do more and communicate in a more fulsome way.

**THE COURT:** Is a human telling it that, or does a computer somehow evaluate whether it's a good or bad answer?

**MR. WINTHROP:** Both. There are humans involved in this pre-training -- in this fine-tuning process, but part of what the humans are doing is also creating models that can judge: Is that a good answer or a bad answer?

So they use smaller models, for example, to -- that are trained on samples of inputs and, you know, good and bad outputs. So that way the model can learn what's a good -- a useful response and a not-so-useful response. And that, like,

helper model, if you will, can help with the fine-tuning of the LLM. So it's a mix of both.

Next topic we have is constitutional AI and guardrails.

So the term "constitutional AI," that's an Anthropic term. That refers to an approach of developing AI systems with certain behavioral constraints and principles built in during training.

So among those constitutional AI principles is avoiding copyright infringement. And I'll show you a slide that -- relating to your question about if you asked it to, you know, print out the first chapter of some copyrighted book.

In addition, LLMs typically have other features that are relevant here. So when a user enters a prompt like, for example, the one that you hypothesized, if that prompt may generate a response that is not desirable, prompt-side filtering is designed to identify those prompts and either redirect or refuse to answer them.

In addition, outside filtering -- output-side filtering is a feature that compares a potential response against copyrighted works to prevent a model from providing copyrighted material in response to a user prompt.

And then this is also something I wanted to note because -- again, from the -- the presentation you heard this morning.

On the relationship between these principles -- so

constitutional AI, prompt-side filtering and output-side filtering and how they relate to basic training, all right, they seem to suggest in the presentation this morning that the goal of an LLM is to mimic the training data; that this whole process was designed to determine or see if you can set up a system where the model can mimic the training data. That is actually not true.

A model that simply repeats training data is not a good model. The repetition of training data is not what makes an AI model a useful assistant.

And so when Anthropic trainers, for example, see this happening, there is a term for it. It's called over-fitting, if the model is too -- adhering too close to training data. The model is actually penalized, a term for meaning the weights are adjusted, right, to prevent that. You don't want that to happen.

And so that the model learns that repeating training data is not a preferred response to a user prompt. So that's one place where we differ from how they were presenting all of this.

**THE COURT:** Can I ask a question?

All the inputs here seem to be words, words. The inputs are words. But somehow I had the idea that pictures, for example, or sounds even, were factored into the database.

So like bird sounds. I have an app that I can tell what

bird it is by listening to the -- and it will tell me. That's a such-and-such, acorn woodpecker.

So does Claude do that? Can Claude listen? And is Claude trained on sounds and music and facial recognition, that kind of thing?

**MR. WINTHROP:** What I can tell you is the reason that we have focused on words, and the reason the plaintiffs have focused on words, is because this case is about books. So, therefore, we have not been focused on your -- what you said, pictures, sounds, that sort of thing.

My understanding is -- is yes, there is an aspect of that kind of training that occurs. But I would not want to represent to you that I have studied that because I have been focused on the words for this case, but I believe you are correct.

**THE COURT:** Okay.

**MR. WINTHROP:** So we -- we mentioned the -- this idea of the constitutional AI and guardrails and prompt-side filtering and output-side filtering, and I said I would give you examples.

So if you go to the next slide.

There. You see this is -- you asked about could the book -- could the model produce in response, you know, a chapter of the book. And this is what you're going to get. This is -- in this prompt it's DW. Those are my initials.

I asked it: Please give me the script of *Harry Potter and the Order of the Phoenix*.

You get a response [as read]:

"I apologize, but I cannot provide the script of *Harry Potter and the Order of the Phoenix* as that would be a copyright violation. The screenplay, like the book, is protected intellectual property, et cetera. If you're interested in studying the film's structure or specific scenes, I can discuss them in general terms or point you to publicly available resources."

They did show you in their presentation a section of *Tale of Two Cities*. Of course, *Tale of Two Cities* is out of copyright; right?

Okay. Last topic is running and using the model, what is referred to as inference.

So here is the big picture. The user sends a prompt to the model for processing. The input text is broken into tokens, and each token is assigned a number that the model can understand and process.

The tokens are further converted into vectors, which capture the mathematical relationships between tokens.

And, finally, this encoding is passed through the model, referred to as the forward pass, is the term that's used, in which matrix multiplication and other processes generate a

probabilistic result.

And here is an example on the screen of what that is, the -- the idea of showing the probabilistic result.

It's very important to understand that in deciding what response that the model will give, the model is not trained to give a precise specific response or a predetermined response or one response. There is a -- there are a range of appropriate responses based on probabilities from what it has -- the training data, what is learned from the training data. And the model is going to use sampling to decide what -- which response to give in any given case.

So if you look at the graphic that we've shown here, in this example, "The city experiences cool summer temperatures due to..." And the question is what's going to be the -- you know, what's it going to -- what's the response going to be?

The model will complete the given statement 33 percent of the time with "marine." 27 percent of the time with "ocean." 14 percent of the time with "geographical." And 8 percent of the time with "the." And then the rest of it to get to 100 percent is smaller numbers.

And this goes back to the notion of known uncertainty. Inherent in the response generated by an LLM is a certain level of uncertainty within the known constraints of this kind of sampling of appropriate responses.

And that's why, when you go to the next slide, this is how

this works in practice. We asked: Write me a Haiku about San Francisco.

You will note that they are different. This is -- this is the same model. This is the same prompt. And I suspect this was done within minutes of each other, seconds of each other. So that there are many possible outputs even for the exact same prompt.

Each output is the result of the process involving the millions of statistical calculations previously discussed that for the same input can provide different outputs.

And so this is unique and interesting. It contrasts with deterministic coding, where you may, for example, right, query a database for a specific value and the same value is always returned. That is not what these models do and that's not how they are set up.

So this is how the technology works.

**THE COURT:** Okay. Just a sec.

(Brief pause.)

**THE COURT:** Could you ask Claude or any of these other programs to summarize all of the news reports for the day into a single memo so that you didn't have to go to *Wall Street Journal, Washington Post*? You could get Claude's version of the morning news? Is that doable?

**MR. WINTHROP:** Not really because the models are trained as of a certain date. And so, for example, if you were

to say to Claude: Who is the President of the United States? Which I asked Claude this last night. As a great surprise to Donald Trump, Claude said: Joe Biden is the president of the United States because -- and then the model goes on to say: Because I was trained as of a certain date.

**THE COURT:** Okay. Let's say a different example. Let's say you were to say for the first 24 years of this decade -- not decade, century, for each year give me one paragraph of the most important things that happened.

**MR. WINTHROP:** I would --

**THE COURT:** Would it be able to do that?

**MR. WINTHROP:** I think probably. I think probably. It may not be perfect, but it might -- I think it could generate something, yeah.

**THE COURT:** If you were to say: What were the best movies? And let Claude decide what the best movies were --

**MR. WINTHROP:** For each year --

**THE COURT:** -- could it do that, or would it give you an answer that makes some sense?

**MR. WINTHROP:** I think it might be able to tell you from what -- of what -- it might say what were the movies that got certain awards or that sort of thing. I think -- I would expect you would get some kind of response.

And one thing that's interesting that I've noticed when you -- so I have also used this -- this tool, is it can

iterate. So, for example, you could say, as I did: Draft a poem in honor of my brother-in-law's birthday, and he's a lawyer, and he's got two kids, and his kids play soccer. And it drafted -- it came up with a poem.

Then I said: Oh, you know, I forgot, like, my sister-in-law is also a lawyer. We should include that. And the kids aren't toddlers, they are actually in high school and college, so I need a little more advanced. And I said that. I typed that into Claude. And it came back and it iterated in that way and had a more -- you know, made the kids older, that sort of thing, and included some reference to the wife.

So it's quite, you know -- it's possible in that way to kind of communicate with Claude or sense of iterate and give it more data, more information to help it come up with appropriate useful response.

**THE COURT:** Let me change the subject just slightly. I don't want to get into the merits too much, but I want to give each of you three sentences; maybe five, but no more than a total of a minute to -- with the benefit of what I've learned here, I want the plaintiff to explain what the copyright violation is. And I want you to explain your view of why it's not. So let's -- let's hear what the one-minute version is.

**MR. NELSON:** Thank you, Your Honor.

As we saw in one of the slides, Anthropic knowingly went to a pirated dataset and downloaded, copied that dataset and

used it, period. That alone -- you can stop there, full stop. That is the *Napster* case. You cannot do that. That is paradigmatic copyright infringement.

Now, we can go on and talk about other copyright infringements. My colleague is 100 percent correct. This is not an output copyright infringement. But the use of training data by itself is also copyright infringement as it goes on.

There is a market emerging that you go through the four steps of fair use. It hits every one of those four steps.

My expectation is that they are going to say it is transformative because the output is transformative, but that is not the test.

As we saw in the training data, it is consistently copying and using that expression to train, number one.

Number two, it is for a commercial use, but it is -- just as the *Andy Warhol* case said, if you go through the other factors, including the emerging licensing market, you will see that it is not fair use. There is no doubt it is prima facie copyright infringement. Putting aside, I think, the clear pirated part of it, even going beyond that, is also copyright infringement.

Thank you, Your Honor.

**THE COURT:** Okay. What's your one-minute version?

**MR. WINTHROP:** This is a quintessential fair use. In every copyright fair use case, there is copying.

**THE COURT:** There is what?

**MR. WINTHROP:** There is copying. That just gets you -- that's the entrance to the discussion about fair use.

The question is: What's it for? What's the use? Is it a different use than what the copyright holder's use and purpose was?

This is fundamental. This is using what -- using works, using to learn the language, to study language. It's not expressive. It's basically extracting data about language and that is not copyright infringement. It is a fair use. This whole notion of --

**THE COURT:** If you went through the four statutory factors --

**MR. WINTHROP:** Fundamentally a completely different change, a different use. Fundamentally different use. Completely transformative use.

**THE COURT:** But at the moment you copy it, it is the exact copy.

**MR. WINTHROP:** In any fair use case there is always some copying, and it's always -- you can often say you start with a copy and then do you something with it.

I don't think that's going to be the answer. And I think if you think of *Napster*, that's their analogy. *Napster* is you copy a song and someone plays a song. They copied the song and they played the song.

We have been here for two hours. There is no notion that Anthropic is generating anything that infringes any copyright. Nothing that they -- they have showed nothing that says anything is substantially similar.

**THE COURT:** What the four statutory factors? It's commercial use --

**MR. WINTHROP:** Amount you --

**THE COURT:** -- the extent of copying. I've forgotten the other two off the top of my head.

**MR. WINTHROP:** It's the impact on the market. It's the extent of your use. How much of their -- their -- the plaintiff's use -- the work was used.

And so when you run through those, they go in different directions and certain elements. And the Courts say some are more important than others.

The most important one is the first one. What is our -- what is the defendant's purpose and use? And are we doing something different? Is it transformative?

And I think this is -- the two hours of presentations should have shown you it's quite different than anything that folks have been talking about.

**THE COURT:** All right. Now, you got more than one minute, so you get a rebuttal.

**MR. NELSON:** Thank you, Your Honor.

When you look at just that copying, they are using it for

the exact purpose. There's is no gloss. There's no derivation even. It is -- they are using it for how the words are put together. They are using it for the human expressive content in the training data.

It is simply not true that you can only look at the output-side and say: Well, nothing that happened in between, the intermediate copying --

**THE COURT:** He's not saying that. What he's saying is, yes, we copied it, but that's the starting point. He's saying that they copied it for a transformative purpose.

**MR. NELSON:** They copied it -- no. They copied it to extract the expressive content and make exact copies during the training process.

And while it is true that -- even if you accept -- which I don't think the case law will show, but even if you accept that you do look at the output-side of it, okay, as the Warhol case just said, you also, according to the statutory language and the case law, you have to look at the commercial use as well.

They are copying swaths of these books. The 196- -- just on Books3, the 196,000 books alone for their expressive content. And then so what are they doing then? They are looking at how it's formed.

And Your Honor asked a question, which I thought was very perceptive -- and I did ask our expert during the break -- and on this next token predictor, next word predictor, how does it

know the plot.

And the answer is it has all of these books, all of this training set that is built into it, and it does it on multiple levels; the chapter, the structure.

And so if you have a small dataset, it doesn't necessarily get you there. But as you expand out, you look, you know what the training set is about. You understand that the -- it is the model that has ingested this. And it is making exact replicas, exact replicas of this.

So the only fair use thing is, okay, well, if you're going to look at the output alone, you know, is that enough to overcome the exact replicas that it is making in the training data for the purpose of having the model understand -- and, by the way, it's not a brain, it's code; okay? -- having the model understand how words are strung together for that contextual encoding and how words are strung together.

**THE COURT:** Well, the -- I see that point, but let me ask you: Let's say you just had two books and you did the weighting, so the -- how the words. In one book it was an 89 weighting and the other book it was 14. So the computer merges them together and comes up with something like 50, and -- which is neither.

So in going through 180,000 books, it's going to come up with an average that represents none of the books. It represents the average of -- some weighted numerical

representation of how often these words appear next to each other, which is not going to -- not going to translate to any of the particular works unless it was just a statistical fluke.

So what would be your answer to that; that the output is different, the output is transformative, even though they do, in fact, literally copy exact words, the entirety of the work?

**MR. NELSON:** Well, I would say that this case is not an output case and they might have a -- something that talks about a defense on the output-side, if it's generating something about the output-side of this.

Our case is specifically about -- just -- let's focus just alone on that initial pirated copying. I'll be corrected if I'm wrong, but I don't know if there's a case or someone has gone to a pirated site and downloaded knowing that that was illegal content and then use that data, forgetting about the output.

Your Honor asked, for example, at the scheduling conference: What if you buy an illegal copy of a book and then write a book review about it? All right? The book review is going to be fair use; right? But that doesn't excuse the fact that you stole the book or that you illegally downloaded the book. The fair use, whatever it is, on the output-side --

**THE COURT:** Okay. That's a good analogy.

What's your answer to that?

**MR. WINTHROP:** I think there's going to be a big

difference here between the parties. What they want to say is that the process of using this data in training is taking the expressive -- the expressive --

**THE COURT:** But he's saying you stole it. You downloaded a pirated copy.

**MR. WINTHROP:** There is ample case law that this whole idea of good faith, bad faith, all of that in fair use is virtually irrelevant. And we will cite all of the cases.

**THE COURT:** Okay. That's not my memory, but --

**MR. WINTHROP:** In many, many, many cases that -- the party that is doing something doesn't have authorization to copy. And the Courts have said that's not -- that doesn't get you -- bar your fair use defense.

**THE COURT:** Okay.

**MR. WINTHROP:** And let me just -- one -- I want to make one point.

**THE COURT:** One last point --

**MR. WINTHROP:** Yes.

**THE COURT:** And then we've got to go.

**MR. WINTHROP:** Is that there -- we disagree fundamentally that what's going on here is extracting the expressive content of these materials. That is not -- and your analogy, I think, of the algorithm kind of demonstrates that. And that's why there is no output that infringes copyright.

**THE COURT:** Okay.

**MR. NELSON:** Can I just for three seconds?

No, no. Sorry.

**THE COURT:** All right. What do you want to say? Fifteen seconds.

**MR. NELSON:** Really briefly, really briefly. The reason why these so-called guardrails are in place is because, of course, it can replicate content. You can summarize the plot of the *Feather Thief* if you ask.

So the fact that the model, they have chosen to put in these constraints --

**THE COURT:** Well, I see that point, but it's not going to do it in every case. It's going to do it in a rare case, and the guardrail is there to guard against the rare case.

I don't think it would -- unless somebody asked: Give me the text for your copyrighted work. Then, of course. So, but in the ordinary case, it's not going to monkey at the typewriter, come up with that book.

**MR. NELSON:** That's right. And that's why it's not an output case, Your Honor.

**THE COURT:** All right. Who helped you do your slides?

**MR. WINTHROP:** I had a lot of help. I had Jessica Gillotte here. My colleague Joe Farris did a lot on the slides. Our client did a lot on the slides. And we had help from FTI over here in the back.

**THE COURT:** Both sides did an excellent job. I thank

you. I've learned a lot today and I look forward to seeing you again.

Okay. Bye-bye.

(Proceedings adjourned.)

**CERTIFICATE OF OFFICIAL REPORTER**

I certify that the foregoing is a correct transcript from the record of proceedings in the above-entitled matter.

Debra L. Pas

Debra L. Pas, CSR 11916, CRR, RMR, RPR

Saturday, February 8, 2025